# Application mobile AriaryMarchand

## Objectifs du projet

L’application mobile a pour objectif de vulgariser les services proposés par Ariary.net, de créer un système de transaction et d’échange le plus simple, rapide et sécurisée possible. Tout cela, en utilisant l’objet le plus attaché à l’homme actuel, son « Smartphone ».

## Planning de réalisation

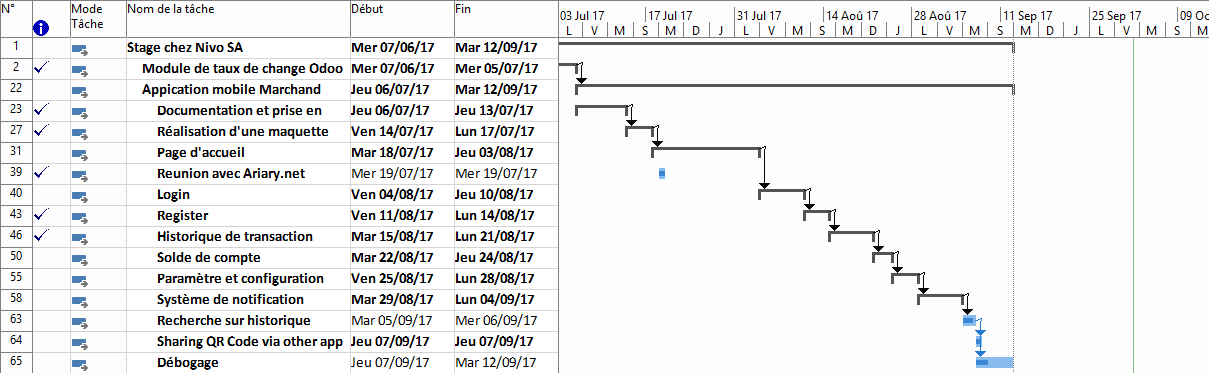


Figure 1: Diagramme de GANTT - Application mobile

Coché : les parties de l’application qui sont effectuées

Non coché : Il y a des composants abandonnés ou en cours de réalisation

## Technologies utilisées

### Le QR Code – Code barre 2D

Une code-barres est manière de présenter sous format numérique des données et des informations qu’on peut voir comme un symbole et qui est lu facilement par des appareils dites lecteurs de code-barres. Ces appareils peuvent être des appareils spécialisés ou simplement des smartphones.

Les codes-barres 2D sont des formes avancées offrants plus de possibilité sur l’utilisation des codes-barres. Si le simple code-barres à une Dimension ne contient de l’information que dans l’horizontal et que sa taille varie selon celle des données qu’elles contiennent, les 2 dimensions permettent d’avoir des données sur les 2 axes avec une taille non dépendante de ses données.

Figure 2 : Différence entre code barre 2D et 1D

Comme notre projet consiste à faire une échange rapide et simple, nous avons opté pour le choix des codes-barres à 2 Dimensions comme solution pour les transactions et plus précisément le QR Code « Quick Response Code », le code barre à 2 Dimensions le plus utilisé dans le monde. Ce choix du QR Code a été renforcé pour les raisons suivantes :

* Il est très facilement et rapidement lu par les smartphones. Cette lecture se fait par l’utilisation d’application dédiée, de l’appareil photo et le tour est joué.
* Sa capacité de stockage de données et largement grande et peuvent contenir jusqu’à 7089 caractères numériques et/ou 4296 caractères alphanumériques.
* Comme le taille du QR Code ne dépend pas de celle de ses données et que c’est en 2 dimensions, sa forme tient parfaitement sur un écran de smartphone.
* Sa forme et l’ajout de personnalisation est disponible, permettant ainsi de faire un QR code bien propre à Ariary.net.



Figure 3 : Code barre personnalisé pour Ariary.net

### Framework React-native

React-native est un Framework JavaScript hybride[[1]](#footnote-1) développé par Facebook pour le développement d’application mobile. Nous n’avons pas hésité à faire ce choix vu les possibilités qu’il offre. En effet, react-native est en premier lieu un framework Open Source, et Hybride. De son slogan « Learn once, write everywhere », il permet de créer une application compatible à tous les systèmes mobiles existants (Android, IOS, Windows Phone, …) avec un seul code source écrit avec JavaScript. Le framework se démarque aussi des autres hybrides (Cordova, Ionic, …) du fait qu’il utilise les composants natives (propres aux systèmes) au lieu d’utiliser des « Webview » pour le rendu de l’application, il offre ainsi une performance bien au-dessus de ses concurrents.

Étant donné qu’il est en JavaScript et Open Source, une large communauté et des libraires externes (propre à react-native et les librairies JavaScripts[[2]](#footnote-2)) sont disponibles avec une facilité d’installation et d’utilisation.

### Expo Project

C’est un ensemble d’outils, de services et de multiples librairies que l’on peut utiliser avec react-native. Il simplifie grandement le code source avec un développement et débogage en temps réel avec l’application Expo installé sur le mobile de test.

Il offre aussi la possibilité d’un système de notification basée sur l’identification du mobile et l’application. Une fonctionnalité indispensable pour notre application.

En utilisant l’Expo XDE[[3]](#footnote-3) (outils de développement) ou simplement la ligne de commande et la librairie expo, on peut générer une application portable (Standalone App) pour un plateforme choisi. Un manière rapide et efficace.

### Le protocole Oauth 2

L’Oauth2 est une deuxième version du framework de sécurité Oauth. Son objectif est de permettre à une autre entité (application) d’avoir accès aux données et à l’API du serveur de façon sécurisée. En d’autre termes, c’est un protocole d’accès à des services web. Cette version en question est actuellement la norme utilisée pour avoir accès aux données des utilisateurs des grands conglomérats du monde de l’informatique (Facebook et Google…). L’objectif de ce protocole est de permettre à un client de parler aux serveurs seulement quand il a un clé d’accès valide du nom Token qui lui a été généré lors de sa connexion. Ce Token est une chaîne de caractère qui sera la seule porte pour avoir des données. Et on a 2 types de Token :

Access Token : une clé permettant d’avoir des données mais d’une durée de validation limitée. Il peut être utilisé plusieurs fois.

Refresh Token : De son nom, il permet d’avoir un nouvel Access Token quand celui que vous utilisé n’est plus valide. On ne peut l’utiliser qu’une seule fois.

Son utilisation permet ainsi de donner des informations et des accès qui, selon de type d’autorisation légué, pourront être utilisé par de différentes entités. On peut créer son propre type d’autorisation mais les bases sont :

**Autorisation Grant Code :** On obtient avec ce type d’autorisation un « Access Token » et un « Refresh Token ». Elle est surtout destinée aux clients dits confidentiels, c’est-à-dire, un client qui demande est un serveur.

**Implicit Grant** **:** Ce type ne permet que d’avoir un « Access Token » après que l’utilisateur se soit connecté sur une redirection. Elle est utilisée dans les applications mobiles tierces et les applications JavaScripts (dans des navigateurs)

**Credentials Grant :** Celle-ci n’est utilisé que lorsque le client et le serveur sont de la même entité, on parle ici d’un produit venant de la même entreprise. Il aura accès à un « Access Token » et un « Refresh Token ».

**Clients Credentials Grant :** Elle permet d’avoir un accès d’usage bien limité et bien défini à un client sur les données. On aura alors un « Access Token » en retour.

### Back-end – PHP

Pour la génération et la gestion des notifications avec Expo, nous avons opté pour PHP par le simple fait que c’est un langage Orienté objet, dynamique. Mais aussi, il est bien adapté pour les webservices RESTful.

### SGBD - MySQL

PostgreSQL et MySQL sont des systèmes de gestion de base de donnée (SGBD) gratuit et libre. Notre but ici est de créer un serveur passerelle entre le vrai serveur d’Ariary.net et l’application mobile. Son but est de mémoriser des données propres à l’application sans devoir faire des modifications au niveau de la base actuelle d’Ariary. Nous avons choisi MySQL par le fait qu’il est surtout destiné pour des petits et moyens volume de données.

# Réalisation de l’application

## Analyse et conception

### Analyse de l'existant

Pour le cas de notre pays, on peut dire que la majorité des transactions se fait toujours avec de l’argent liquide (espèce). Un procédé qui possède de multiples inconvénients surtout l’insécurité et l’arrondissement des prix.

D’autres formes d’échange sont aussi disponible dans notre pays tels que les virements bancaires ou encore l’utilisation des Mobile Money. Mais leurs inconvénients résident soit dans ses procédures qui sont longues et avec beaucoup de formalité soit sur un fonctionnement compliqué et qui peut durer plusieurs minutes (trop de temps perdu si on veut juste acheter rapidement du jus ou autres produits de consommation dans une épicerie).

C’est pour ces raisons que nous avons mises en place notre système qui assure l’échange et payement en un temps record sans fléchir en terme de fiabilité et de sécurité.

### Conception de l’application

Rappelons tout d’abord qu’avant la création de cette application mobile, le système Ariary.net était déjà opérationnel. Il avait déjà ses propres services et ses propres sécurités mais ne suivait pas encore les normes du protocole Oauth2.

En prenant cela en compte, nous avons procédé à la conception de l’application de la manière suivante :

**Recherche d’application similaire et maquette** : En se souvenant d’une phrase que l’on nous a dit en deuxième année « On ne réinvente pas la roue » et du fait que déjà bon nombre de personne ont fait la même chose que nous auparavant, nous avons fait des recherches pour trouver des applications qui font les mêmes choses et qui ont les mêmes ambitions que nous. Et c’est ainsi que l’on a trouvé 2 applications de référence :

* **BlockChain** : application mobile pour Bitcoin basée aussi sur le QR Code (très similaire à Ariary.net)
* **Paypal** : l’une des solutions de paiement de plus utilisé dans le monde.

C’est en basant sur ces 2 applications que nous avons créées les maquettes d’Ariary.net mobile. Ce n’est pas une question de Copier-Coller mais une sorte de référence et une projection vers notre finalité.

**Création d’un serveur de test :** Comme dit ci-dessus, le service ariary.net, déjà implémenté et utilisé, ne suivait pas encore la norme du protocole Oauth2, nous avons mis en place un serveur de test qui lui respectait cette norme. Nous avons fait ce choix pour ne pas perdre du temps en attendant le changement au sein du principal serveur. Ce serveur de test ne comportait que les stricts minimums pour la réalisation d’une transaction pour ne pas s’attarder à des faux problèmes. De ce fait, nous n’avons eu qu’une seule table « transaction » permettant à la fois de faire une transaction, de calculer le solde et de créer un compte en insérant une transaction à montant 0.

**Mise en place des fonctionnalités de base :** Pour faire bien avancer le projet, nous avons fait en sorte de créer une application, avec comme seule fonctionnalité la génération du QR Code et la transaction monétaire, en corolaire avec la création du serveur de Test. Ceci permet ainsi d’avoir un aperçu de la future application. La transaction en question ne comportait encore aucun système de sécurité.

**Création des fonctionnalités complémentaires et amélioration des contenus :** c’est seulement après cela que toutes les autres fonctionnalités comme la connexion, historique, paramètres et autres ont été mis en place.

**Système de notification :** Cette partie du projet consistait à modifier, d’ajouter au serveur de Test des services permettant de construire un système de notification et d’implémenter les fonctionnalités qui y corresponde au sein de l’application. Ainsi, on a ajouté une table « account » qui sauvegarde les données d’un utilisateur en rapport avec le téléphone qu’il utilise. Cette table permet d’avoir une liste et un suivi des mobiles qui sont connectées au compte d’un utilisateur.

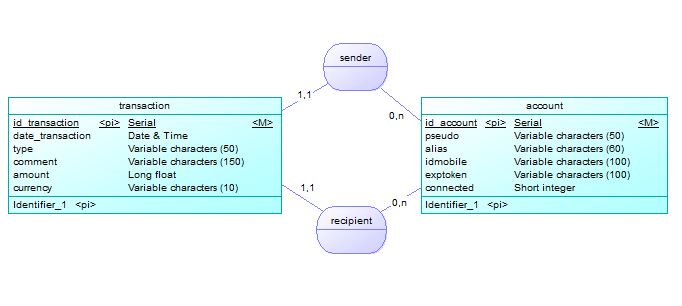


Figure 4 : Les tables du serveur de Test

**Ajout du norme Oauth2 :** C’est seulement à cette étape du projet que l’on a implémenté dans chaque service le protocole Oauth2 assurant ainsi les échanges avec le serveur.

**Passer du serveur de test au relais :** Durant le développement de l’application nous avons choisi de transformer le serveur de Test en une passerelle reliant le serveur d’Ariary.net et l’application mobile. Avec cela, on a la possibilité de sauvegarder des données propres à l’application (surtout dans le domaine de la notification et du suivi) sans avoir à toucher les données du serveur principal. Ainsi, chaque service de l’application passe d’abord dans une passerelle avant d’aller vers Ariary.net. Et au retour, s’il y a des actions à faire comme notifier et ou autre, ce sera un travail de la passerelle.

[Diagramme de séquence]

## Développement par fonctionnalité ou module

### Réception d’argent par QR Code

Cette partie du projet est l’un des plus important de cette application car les transactions et les échanges en dépendent complètement.

Elle consiste entre autre à générer et d’afficher un QR code contenant tous les informations permettant de faire les transactions. Ces informations sont :

* L’identifiant du destinataire (ex : AA012)
* Son nom d’utilisateur (ex : Toavina)
* Le montant de la transaction (facultatif)
* L’unité monétaire
* Le type d’échange (ex : trans 🡪 pour désigner un échange de type transaction)

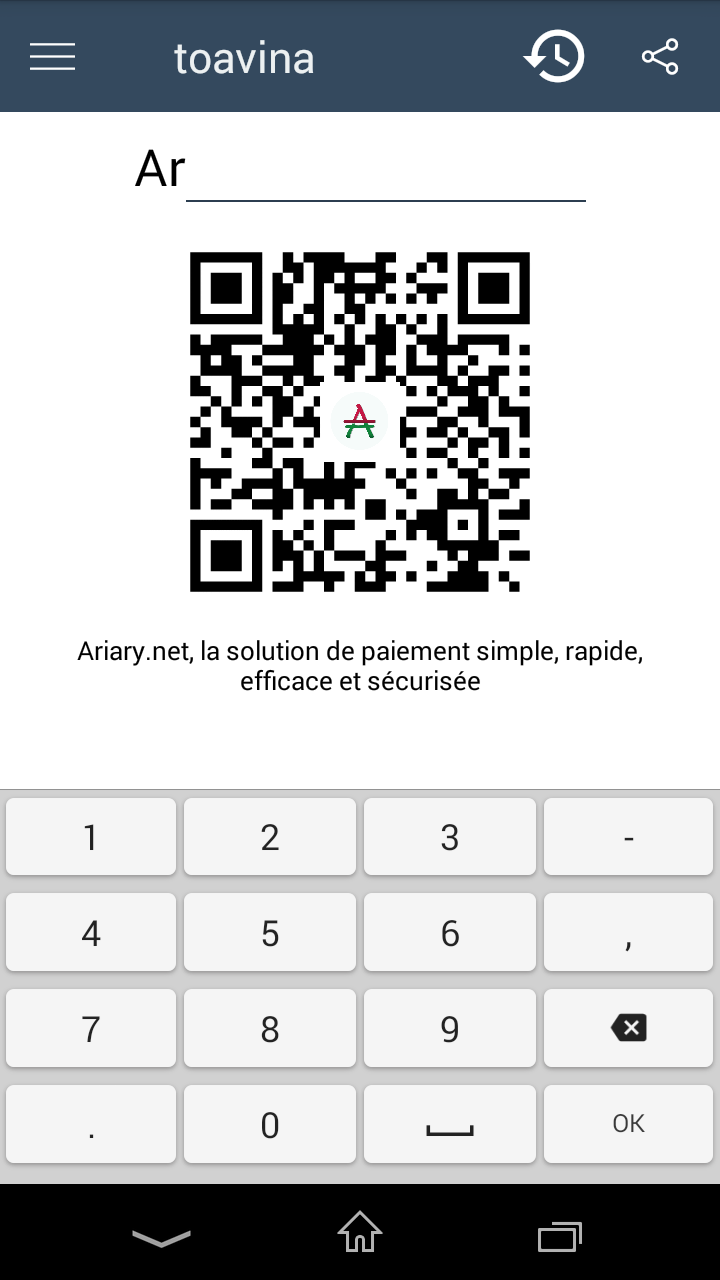
Pour faciliter l’utilisation de l’application, nous avons mis un champ permettant de définir le montant du transfert, il est initialement à zéro (0) et un changement en temps réel du QR Code est visible à l’œil nu quand on ajoute ou modifie le montant.

Figure 5 : Code barre montant

Figure 6 : Code barre sans montant

Pour l’utilisation de cette fonctionnalité, nous avons fait en sorte de faire une interface la plus facile à utiliser. Dès l’arrivée sur la page, l’utilisateur peut déjà recevoir de l’argent mais peut aussi spécifier le montant qu’il veut recevoir. Si le montant n’est pas défini, ce sera alors celui qui envoi qui le fera.

Pour faire la transaction, un autre utilisateur de l’application prend juste le QR Code en photo et on n’a plus qu’attendre. On aura alors une notification à la fin pour confirmer que la transaction a bien aboutit.

[Diagramme de séquence]

### Système de notification – Fiabilité et Sécurité

Une notification est un signal ou une alerte permettant de dire ou de rappeler à un utilisateur d’une action qui vient de se produire, d’un changement ou bien le début ou l’aboutissement d’une activité.

Dans notre application, comme la principale action est la transaction monétaire, l’activité se base surtout sur la fiabilité et la sécurité du système. Dans cette question de fiabilité, on vise surtout à rassurer l’utilisateur sur l’aboutissement de la transaction qu’il a effectué. Pour faire cela, nous avons créé un système permettant d’alerter « toutes les mobiles connectés » au compte de l’utilisateur pour lui confirmer le succès et le type de transaction qui s’effectué depuis son compte.

Pour la bonne pratique, chaque mobile lié aux comptes reçoit la notification même si l’application Ariary.net n’est pas en cours d’exécution. Et pour ne pas faire les choses à moitié, nous avons mis 3 types de notification pour donner plus d’assurance à l’utilisateur :

**Notification en arrière-plan :** Ce type de notification marche même si l’application n’est pas en premier plan. Lors de l’aboutissement d’une transaction, une notification du genre « Vous avez envoyé 2000 Ar à Toavina » ou du style « Vous avez reçu 2000Ar de Hasina » apparait dans la zone de notification du téléphone. Elle est surtout utile pour vous avertir d’une transaction à votre insu.

**Notification rapide :** Celle-ci apparaîtra sur votre écran quand vous êtes dans l’application pour vous dire rapidement qu’un transfert vient d’être effectué. Elle apparaitra seulement pour quelques secondes pour ne pas perturber l’utilisation de votre téléphone. Elle épargnera le temps de devoir rechercher dans la zone de notification si l’alerte qui vient de produire est vraiment la confirmation d’une transaction ou bien autres choses.

**Badge de numérotation** : Il permet de notifier l’utilisateur en affichant sur son écran le nombre de transaction qu’il n’a pas encore vu. Nous avons fait en sorte qu’il soit de couleur bien tape-œil pour qu’on ne le rate pas.

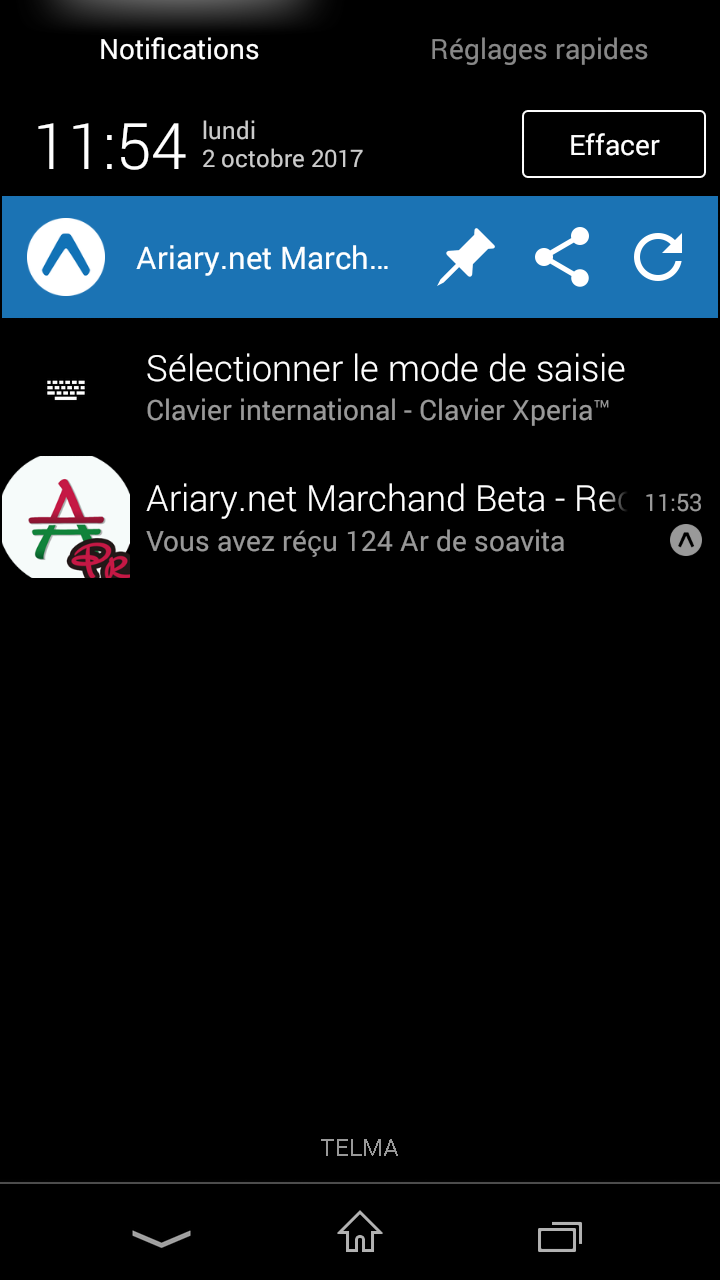
****

Figure 7 : notification en arrière-plan

Figure 8 : Badge de numérotation

Figure 9 : notification rapide

Notre système de notification marche grâce à la création d’un identifiant propre au compte dans le téléphone. Avec cet identifiant, on peut lui faire signe quand une transaction vient d’aboutir sans qu’il soit en écoute. On appelle ce système du « Push notification »

L’identification en question se crée à partir d’une librairie d’Expo permettant de générer une unique clé permettant d’envoyer une notification au mobile. La génération se base sur la combinaison de l’identifiant du téléphone et de l’application Ariary.net.

On aura donc un identifiant de ce genre :

ExponentPushToken[83D9aJCoDivkqZzPSKL7v8]

Cet identifiant est généré lors de la première utilisation et connexion à l’application. Elle sera ensuite envoyée dans le serveur pour la sauvegarde et son utilisation lors d’un envoi de notification.

Le système de notification suit les schémas suivants :

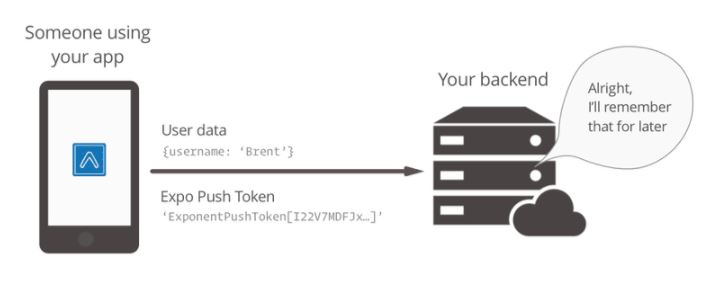
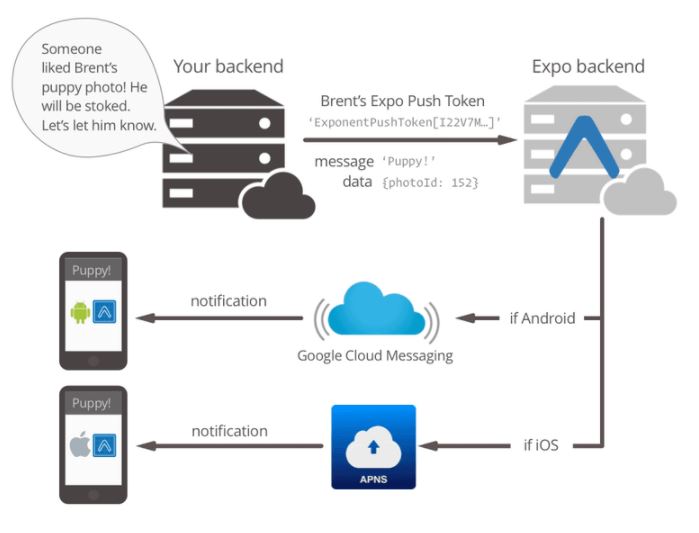


Figure 10 : génération de l'identifiant et sauvegarde dans le serveur



**Figure 11 :** Notification depuis le serveur

**Source :** https://docs.expo.io/versions/latest/guides/push-notifications.html

### Historique de transaction et solde

Pour renforcer la fiabilité citée ci-dessus, un système de suivi des transactions et un affichage des soldes sont implémentés dans l’application.

On a fait en sorte que le solde du compte soit accessible à n’importe quelle partie de l’application. De ce fait, on l’a mis sur le menu de l’application permettant ainsi une vision rapide.

A chaque nouvelle notification, le solde se met à jour automatiquement mais le mode manuel est aussi disponible si besoin.

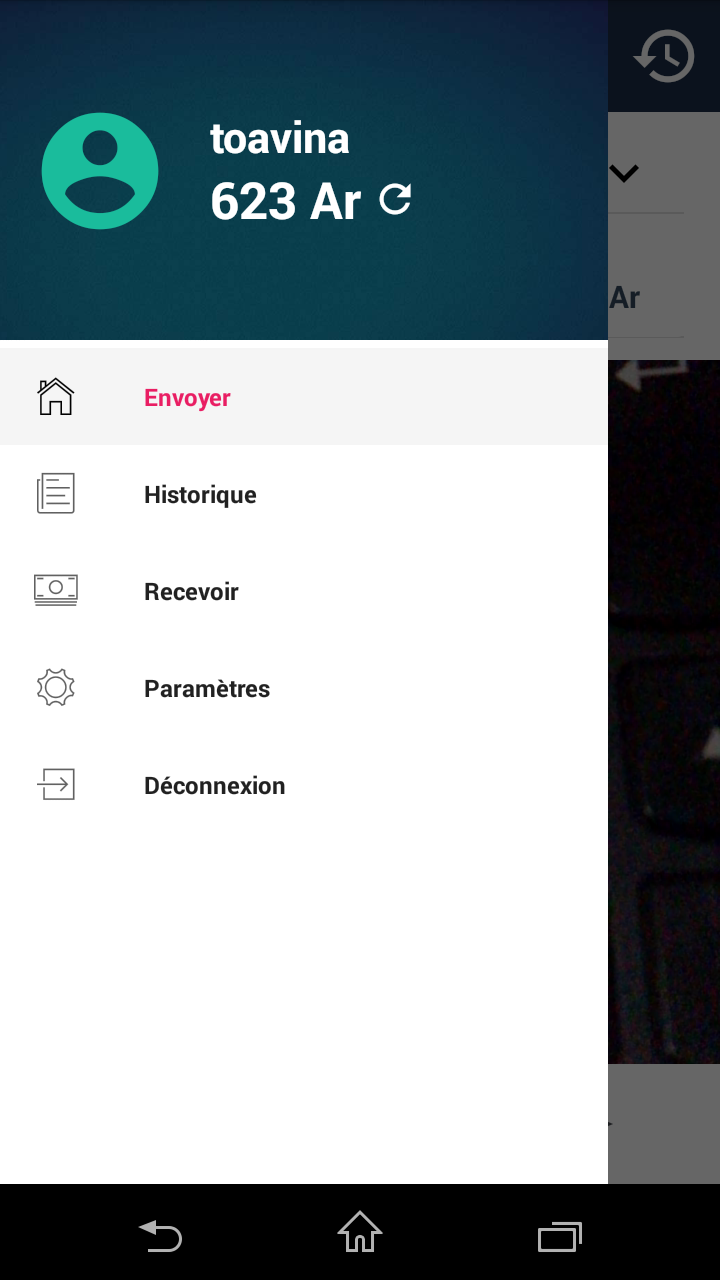


Figure 12 : Solde de compte et liste des menus

Dans un système d’échange monétaire, qu’est-ce qu’il y a de mieux qu’une liste de toutes les actions effectuées pour avoir la confiance d’un utilisateur. C’est dans cette mentalité que nous avons mise en place l’historique de toutes les transactions. Cette fonctionnalité permet un bon suivi des activités et on peut y avoir accès n’importe où dans l’application. (Sur la liste des menus et dans chaque page disponible).

Pour faciliter l’expérience utilisateur, nous avons fait des améliorations pour ne pas avoir une liste monotone :

* Regroupement de la liste par date (permettant ainsi de se situer très rapidement)
* Onglet regroupant les données par type de transaction (entrant ou sortant)
* Une recherche rapide par mot clé.

[Image d’illustration]

### Système de sécurité

Hormis la question de fiabilité, on entend surtout parler de sécurité lorsqu’on rentre dans le domaine de la transaction monétaire. En suivant la norme du protocole de sécurité Oauth2, on a implémenté un système que l’on considère le plus sécurisé actuellement.

De ce fait, depuis la connexion d’un utilisateur jusqu’à chaque requête vers le serveur, nous ne sommes pas sortis de ce cadre et respecte chaque règle de sécurité.

Si nous parlons de la connexion d’utilisateur, le schéma des actions se déroule comme suit :

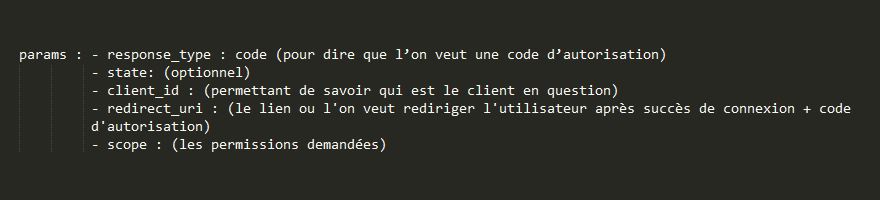
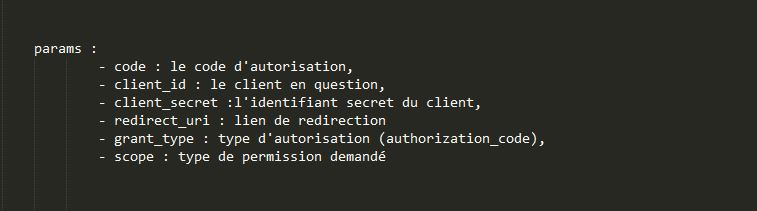
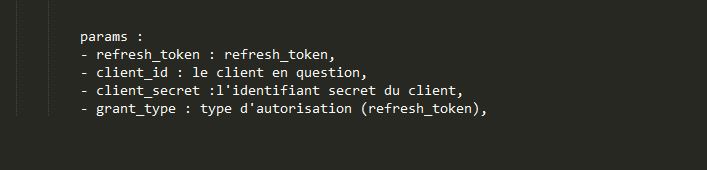
* + Dès qu’il veut se connecter, on redirige l’utilisateur vers la page de connexion d’Ariary.net.
  + Quand il a fini de se connecter, une « code d’autorisation » est générée et placée dans le lien (utilisable en une seule fois).

Figure 13 : Paramètres pour le code d'autorisation

* + Avec ce code d’autorisation, on peut ainsi demander un « Access Token » avec un « Refresh Token » permettant de faire les requêtes.
  + Et c’est enfin après que l’on peut avoir accès à des données avec l’Access Token (données de base de l’utilisateur, solde, historique…).
  + Le Refresh Token quant à lui sert à demander un nouvel Access Token si l’ancien n’est plus valide (utilisable une seule fois)

Voici une vision globale de l’utilisation de l’Oauth2 de type « Authorisation\_Code » sur un schéma :

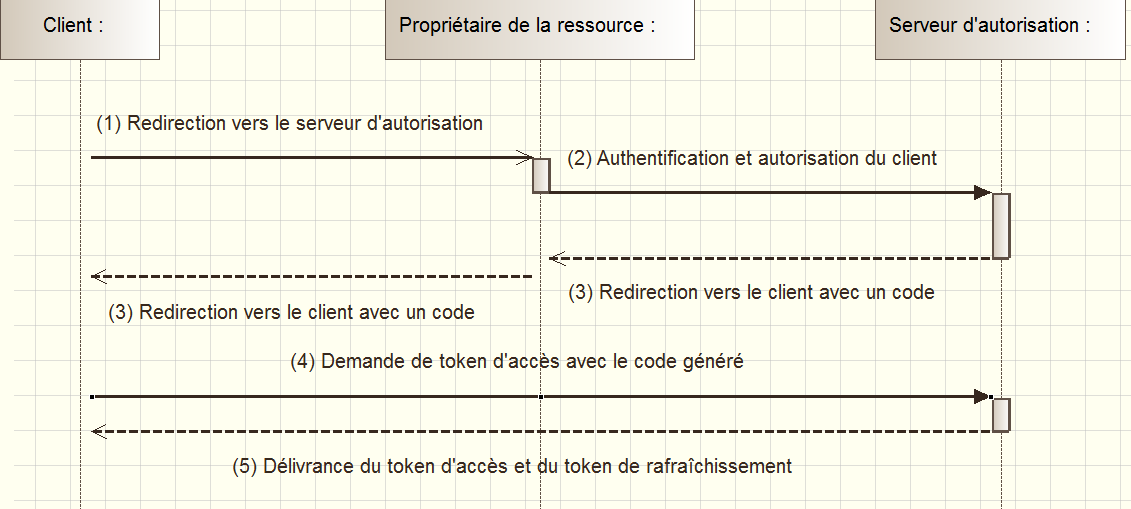


Figure 14 : Système Oauth2 - Type code d'autorisation

Source : <https://zestedesavoir.com/articles/1616/comprendre-oauth-2-0-par-lexemple/#2-les-roles>

### Autres fonctionnalités

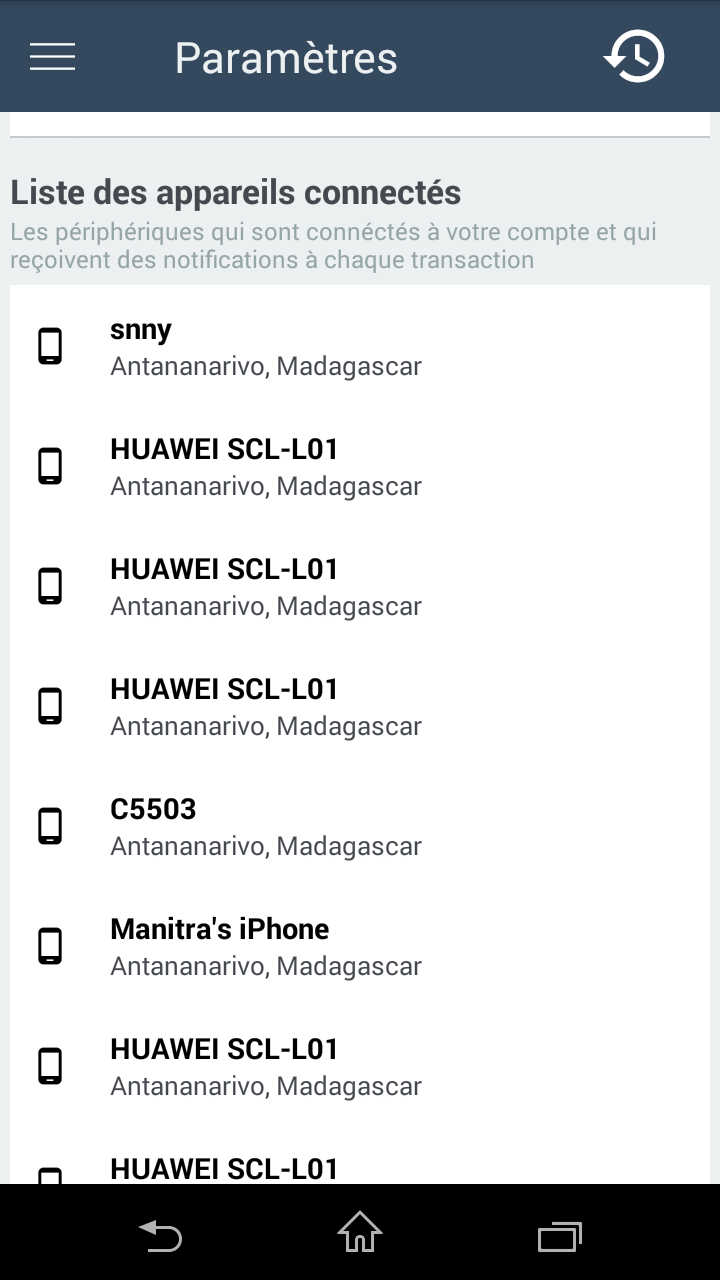
Toujours dans la question de sécurité, on peut trouver dans les paramètres la liste complète des appareils mobiles connectés à votre compte.

Figure 15 : Liste des appareils connectés

## État d’Analyse et Statistiques

### État numéro 1

### État numéro 2

### Statistique numéro 1

Tableau 1 : Classement TIOBE 2014 de quelques langages de programmation

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Position Jan ‘14 | Position Jan ‘13 | Language | Share Jan ‘14 | Trend |
| 1 | 1 | Java | 26.2% | -0.6% |
| 2 | 2 | PhP | 13.2% | -1.6% |
| 3 | 6 | Python | 10.2% | +1.3% |
| 4 | 3 | C# | 9.6% | -0.4% |
| 5 | 4 | C++ | 8.9% | 0% |
| 6 | 5 | C | 8.1% | -0.2% |
| 7 | 7 | JavaScript | 7.6% | +0.3% |

# Bibliographie

DELLEY, A., FRANCIOLI, M., ZBINDEN, P., *Technologies d’accès aux réseaux*, Fribourg : Ecole d’ingénieurs et d’architectes de Fribourg, 2007. 220p.

LALITTE, E., GUICHARD, R., *Apprenez le fonctionnement des réseaux TCP/IP* [en ligne]. Disponible sur : <http://fr.openclassrooms.com/informatique/cours/apprenez-le-fonctionnement-des-reseaux-tcp-ip> (consultée le 28-08-2014)

GAUTHIER C. *Contribution à l'étude du fractionnement de l'aluminium libéré dans des solutions de sols forestiers : influence de la quantité et de la nature de la matière organique*. Thèse de doctorat d’université. Limoges : Université de Limoges, 2003.

<https://zestedesavoir.com/articles/1616/comprendre-oauth-2-0-par-lexemple/>

<https://blog.netapsys.fr/oauth-comment-ca-marche/comment-page-1/>

<https://fr.wikipedia.org/wiki/OAuth>

<http://www.bubblecode.net/fr/2016/01/22/comprendre-oauth2/>

1. Source : https://www.safaribooksonline.com/library/view/learning-react-native/9781491929049/ch01.html [↑](#footnote-ref-1)
2. Exemple : moment, axios, lodash, … [↑](#footnote-ref-2)
3. Expo XDE : [↑](#footnote-ref-3)